RAMP

Research Accelerator for Multiple Processors

David Patterson (Berkeley, CO-PI), Arvind (MIT), Krste Asanovíc (MIT), Derek Chiou (Texas), James Hoe (CMU), Christos Kozyrakis (Stanford), Shih-Lien Lu (Intel), Mark Oskin (Washington), Jan Rabaey (Berkeley), and John Wawrzynek (Berkeley-PI)

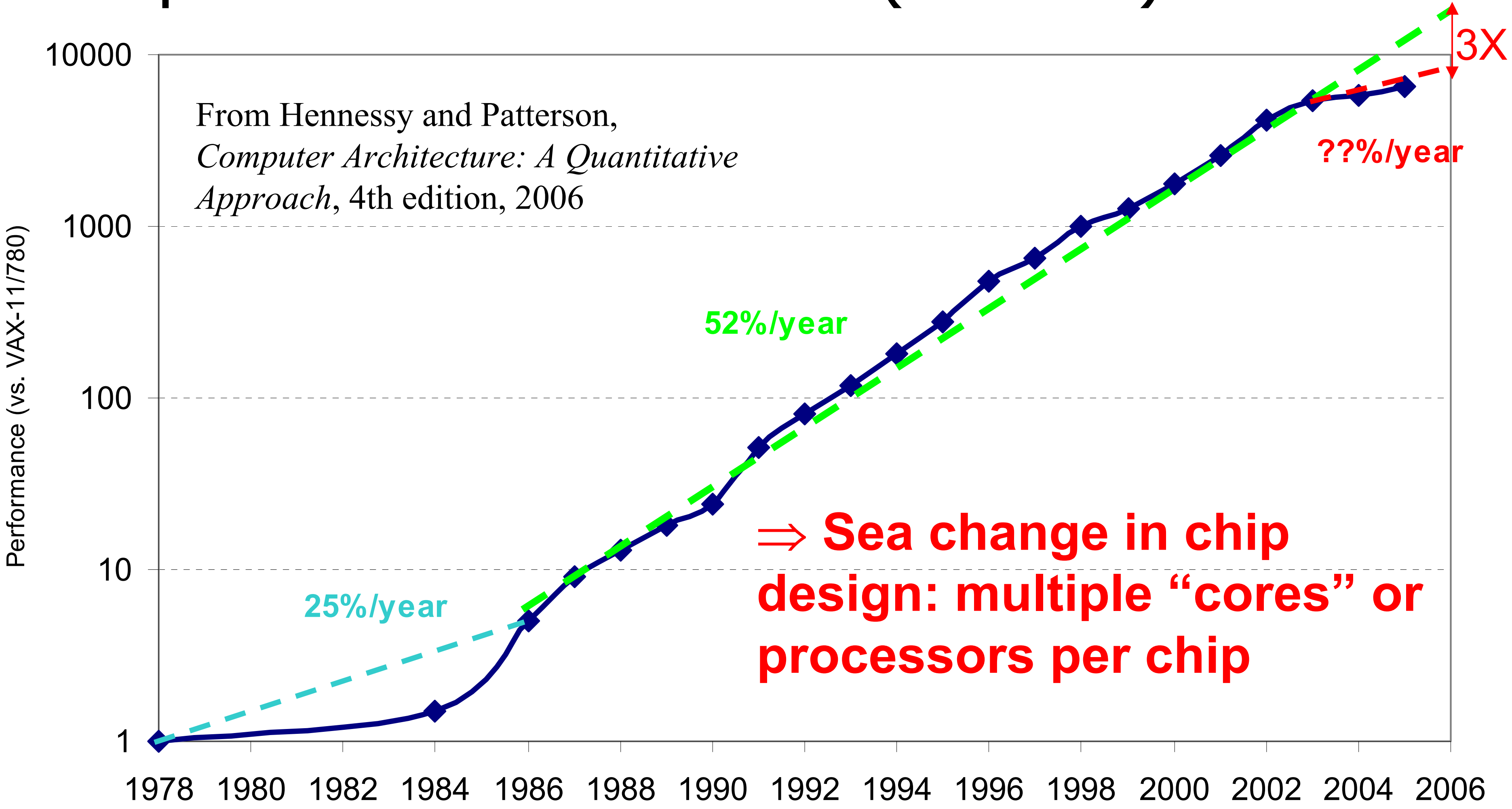
Conventional Wisdom (CW) in Computer Architecture

- Old Conventional Wisdom:
  Demonstrate new ideas by building chips
- New Conventional Wisdom:
  Mask costs, ECAD costs, GHz clock rates mean
  ≈ researchers cannot build believable prototypes
  ⇒ simulation only practical outlet

Conventional Wisdom (CW) in Computer Architecture

- Old CW: Power is free, Transistors expensive
- New CW: "Power wall" Power expensive, Xtors free (Can put more on chip than can afford to turn on)
- Old: Multiplies are slow, Memory access is fast
- New: "Memory wall" Memory slow, multiplies fast (200 clocks to DRAM memory, 4 clocks for FP multiply)
- Old : Increasing Instruction Level Parallelism via compilers, innovation (Out-of-order, speculation, VLIW, …)
- New: "ILP wall" diminishing returns on more ILP HW
- New: Power Wall + Memory Wall + ILP Wall = Brick Wall
  - Old CW: Uniprocessor performance 2X / 1.5 yrs
  - New CW: Uniprocessor performance only 2X / 5 yrs?

Uniprocessor Performance (SPECint)



From Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 4th edition, 2006

⇒ **Sea change in chip design: multiple “cores” or processors per chip**

- **VAX : 25%/year 1978 to 1986**
- **RISC + x86: 52%/year 1986 to 2002**
- **RISC + x86: ??%/year 2002 to present**

Déjà vu all over again?

"… today’s processors … are nearing an impasse as technologies approach the speed of light.."

David Mitchell, *The Transputer: The Time Is Now* (1989)

- Transputer had bad timing (Uniprocessor performance↑)
  ⇒ Procrastination rewarded: 2X seq. perf. / 1.5 years
- "We are dedicating all of our future product development to multicore designs. … This is a sea change in computing"

  Paul Otellini, President, Intel (2005)
- All microprocessor companies switch to MP (2X CPUs / 2 yrs)
  ⇒ Procrastination penalized: 2X sequential perf. / 5 yrs

Manufacturer/Year	AMD/’05	Intel/’06	IBM/’04	Sun/’05
Processors/chip	2	2	2	8
Threads/Processor	1	2	2	4
Threads/chip	2	4	4	32

Problems with “Manycore” Sea Change

1. Algorithms, Programming Languages, Compilers, Operating Systems, Architectures, Libraries, … not ready for 1000 CPUs / chip
2. ≈ Only companies can build HW, and it takes years
3. Software people don’t start working hard until hardware arrives
    - 3 months after HW arrives, SW people list everything that must be fixed, then we all wait 4 years for next iteration of HW/SW
4. How get 1000 CPU systems in hands of researchers to innovate in timely fashion on in algorithms, compilers, languages, OS, architectures, … ?
5. Can avoid waiting years between HW/SW iterations?

Outline

- The Parallel Revolution has started
- RAMP Vision
- RAMP Hardware
- Status and Development Plan
- Description Language
- Related Approaches
- Potential to Accelerate MP&NonMP Research
- Conclusions

Build Academic MPP from FPGAs

- As ≈ 25 CPUs will fit in Field Programmable Gate Array (FPGA), 1000-CPU system from ≈ 40 FPGAs?
  - 8-16 32-bit simple "soft core" RISC at 100MHz in 2004 (Virtex-II)
  - FPGA generations every 1.5 yrs; ≈ 2X CPUs, ≈ 1.2X clock rate
- HW research community does logic design ("gate shareware") to create out-of-the-box, MPP
  - □ E.g., 1000 processor, standard ISA binary-compatible, 64-bit, cache-coherent supercomputer @ ≈ 200 MHz/CPU in 2007
  - □ RAMPants: Arvind (MIT), Krste Asanovíc (MIT), Derek Chiou (Texas), James Hoe (CMU), Christos Kozyrakis (Stanford), Shih-Lien Lu (Intel), Mark Oskin (Washington), David Patterson (Berkeley, Co-PI), Jan Rabaey (Berkeley), and John Wawrzynek (Berkeley, PI)
- "Research Accelerator for Multiple Processors"

Characteristics of Ideal Academic CS Research Parallel Processor?

- Scales – Hard problems at 1000 CPUs
- Cheap to buy – Limited academic research $
- Cheap to operate, Small, Low Power – $ again
- Community – Share SW, training, ideas, …
- Simplifies debugging – High SW churn rate
- Reconfigurable – Test many parameters, imitate many ISAs, many organizations, …
- Credible – Results translate to real computers
- Performance – Fast enough to run real OS and full apps, get results overnight

Why RAMP Good for Research MPP?

RAMP

	SMP	Cluster	Simulate	RAMP
Scalability (1k CPUs)	C	A	A	A
Cost (1k CPUs)	F ($40M)	C ($2-3M)	A+ ($0M)	A ($0.1-0.2M)
Cost of ownership	A	D	A	A
Power/Space (kilowatts, racks)	D (120 kw, 12 racks)	D (120 kw, 12 racks)	A+ (.1 kw, 0.1 racks)	A (1.5 kw, 0.3 racks)
Community	D	A	A	A
Observability	D	C	A+	A+
Reproducibility	B	D	A+	A+
Reconfigurability	D	C	A+	A+
Credibility	A+	A+	F	B+/A-
Perform. (clock)	A (2 GHz)	A (3 GHz)	F (0 GHz)	C (0.1-.2 GHz)
GPA	C	B-	B	A-

Can RAMP keep up?

- FGPA generations: 2X CPUs / 18 months
  - 2X CPUs / 24 months for desktop microprocessors
- 1.1X to 1.3X performance / 18 months
  - 1.2X? / year per CPU on desktop?
- However, goal for RAMP is accurate system emulation, not to be the real system
  - Goal is accurate target performance, parameterized reconfiguration, extensive monitoring, reproducibility, cheap (like a simulator) while being credible and fast enough to emulate 1000s of OS and apps in parallel (like hardware)
  - OK if 20X slower than real 1000 processor hardware, provided >1000X faster than simulator of 1000 CPUs

Accurate Clock Cycle Accounting

- Key to RAMP success is cycle-accurate emulation of parameterized target design
  - As vary number of CPUs, CPU clock rate, cache size and organization, memory latency & BW, interconnet latency & BW, disk latency & BW, Network Interface Card latency & BW, …
  - Least common divisor time unit to drive emulation?
- For research results to be credible
- To run standard, shrink-wrapped OS, DB,…
  - Otherwise fake interrupt times since devices relatively too fast

⇒ Good clock cycle accounting is high priority RAMP project

Why 1000 Processors?

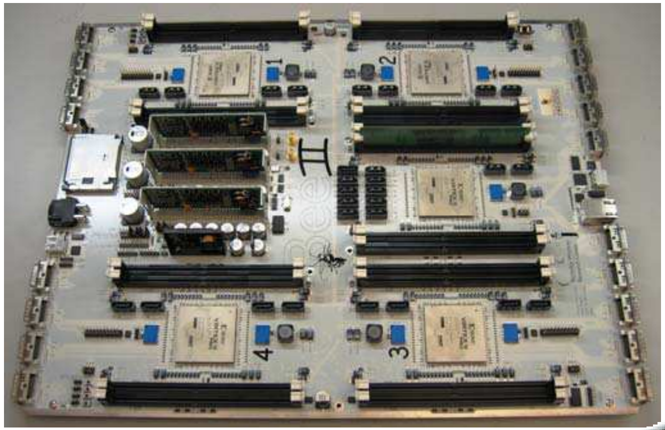
- Eventually can build 1000 processors per chip
- Experience of high performance community on stress of level of parallelism on architectures and algorithms
  - □ 32-way: anything goes
  - □ 100-way: good architecture and bad algorithms
    or bad architecture and good algorithms
  - □ 1000-way: good architecture and good algorithms
- Must solve hard problems to scale to 1000
- Future is promising if can scale to 1000

RAMP 1 Hardware

- Completed Dec. 2004 (14x17 inch 22-layer PCB)

Board:

5 Virtex II FPGAs, 18 banks DDR2-400 memory,
20 10GigE conn.



1.5W / computer,
5 cu. in. /computer,
$100 / computer

Box:

8 compute modules in 8U rack mount chassis

1000 CPUs :
≈1.5 KW,
≈ ¼ rack,
≈ $100,000

BEE2: Berkeley Emulation Engine 2

By John Wawrzynek and Bob Brodersen with students Chen Chang and Pierre Droz

RAMP Storage

- RAMP can emulate disks as well as CPUs
  - □ Inspired by Xen, VMware Virtual Disk models
  - □ Have parameters to act like real disks
  - □ Can emulate performance, but need storage capacity
- Low cost Network Attached Storage to hold emulated disk content
  - □ Use file system on NAS box
  - □ E.g., Sun Fire X4500 Server ("Thumper")
    48 SATA disk drives,
    24TB of storage @ <$2k/TB

4 Rack Units High

the stone soup of architecture research platforms

Wawrzynek	Hardware
Chiou	Glue-support
Patterson	I/O
Hoe	Coherence
Kozyrakis	Monitoring
Asanovic	Cache
Oskin	Net Switch
Arvind	PPC
Lu	x86

Handicapping ISA Donations

- Got it: IBM Power 405 (32b), Sun SPARC v8 (32b), Xilinx Microblaze (32b)
- Sun announced 3/21/06 donating T1 ("Niagara") 64b SPARC to RAMP
- Likely: IBM Power 64b
- Likely: Tensilica
- Probably (haven't asked): MIPS32, MIPS64
- ?? : ARM
- No: x86, x86-64
  - But Derek Chiou of UT looking at x86 binary translation

Quick Sanity Check

- BEE2 4 banks DDR2-400 per FPGA
- Memory BW/FPGA = 4 * 400 * 8B = 12,800 MB/s
- 16 32-bit Microblazes per Virtex II FPGA (last generation)
  - □ Assume 150 MHz, CPI is 1.5 (4-stage pipeline), 33% Load/Stores
  - □ BW need/CPU = 150/1.5 * (1+ 0.33) * 4B ≈ 530 MB/sec
- BW need/FPGA ≈ 16 * 530 ≈ 8500 MB/s
  - □ 2/3 Peak Memory BW / FPGA
- Suppose add caches (.75MB ⇒ 32KI$, 16D$/CPU)
  - □ SPECint2000 I$ Miss 0.5%, D$ Miss 2.8%, 33% Load/stores, 64B blocks*
  - □ BW/CPU = 150/1.5*(0.5% + 33%*2.8%)*64 ≈ 100 MB/s
- BW/FPGA with caches ≈ 16 * 100 MB/s ≈ 1600 MB/s
  - □ 1/8 Peak Memory BW/FPGA; plenty BW available for tracing, …
- Example of optimization to improve emulation

* Cantin and Hill, "Cache Performance for SPEC CPU2000 Benchmarks"

Outline

- Parallel Revolution has started
- RAMP Vision
- RAMP Hardware
- Status and Development Plan
- Description Language
- Related Approaches
- Potential to Accelerate MP&NonMP Research
- Conclusions

3 Examples of RAMP to Inspire Others

- Transactional Memory RAMP
  - Based on Stanford TCC
  - Led by Kozyrakis at Stanford
- Message Passing RAMP
  - First NAS benchmarks (MPI), then Internet Services (LAMP)
  - Led by Patterson and Wawrzynek at Berkeley
- Cache Coherent RAMP
  - Shared memory/Cache coherent (ring-based)
  - Led by Chiou of Texas and Hoe of CMU
- Exercise common RAMP infrastructure
  - RDL, same processor, same OS, same benchmarks, …

RAMP Philosophy

- Build vanilla out-of-the-box examples to attract software community
  - Multiple industrial ISAs, real industrial operating systems, 1000 processors, accurate clock cycle accounting, reproducible, traceable, parameterizable, cheap to buy and operate, …
- But RAMPants have grander plans (will share)
  - Data flow computer (“Wavescalar”) – Oskin @ U. Washington
  - 1,000,000-way MP (“Transactors”) – Asanovic @ MIT
  - Distributed Data Centers (“RAD Lab”) – Patterson @ Berkeley
  - Transactional Memory (“TCC”) – Kozyrakis @ Stanford
  - Reliable Multiprocessors (“PROTOFLEX”) – Hoe @ CMU
  - X86 emulation (“UT FAST”) – Chiou @ Texas
  - Signal Processing in FPGAs (“BEE2”) – Wawrzynek @ Berkeley

RAMP Milestones

- September 2006 Decide on 1st ISA
  - Verification suite, Running full Linux, Size of design (LUTs/BRAMs)
  - Executes comm. app binaries, Configurability, Friendly licensing
- January 2007 milestones for all 3 RAMP examples
  - Run on Xilinx Virtex 2 XUP board
  - Run on 8 RAMP 1 (BEE2) boards
  - 64 to 128 processors
- June 2007 milestones for all 3 RAMPs
  - Accurate clock cycle accounting, I/O model
  - Run on 16 RAMP 1 (BEE2) board and Virtex 5 XUP board
  - 128 to 256 processors
- 2H07: RAMP 2.0 boards on Virtex 5
  - 3rd party sells board, download software and gateware from website on RAMP 2.0 or Xilinx V5 XUP boards

Transactional Memory status (7/06)

- 8 CPUs with 32KB L1 data-cache with Transactional Memory support
  - CPUs are hardcoded PowerPC405, Emulated FPU
  - UMA access to shared memory (no L2 yet)
  - Caches and memory operate at 100MHz
  - Links between FPGAs run at 200MHz
  - CPUs operate at 300MHz
- A separate, 9th, processor runs OS (PowerPC Linux)
- It works: runs SPLASH-2 benchmarks, AI apps, C-version of SpecJBB2000 (3-tier-like benchmark)
- Transactional Memory RAMP runs 100x faster than simulator running on a Apple 2GHz G5 (PowerPC)

Message Passing status (7/06)

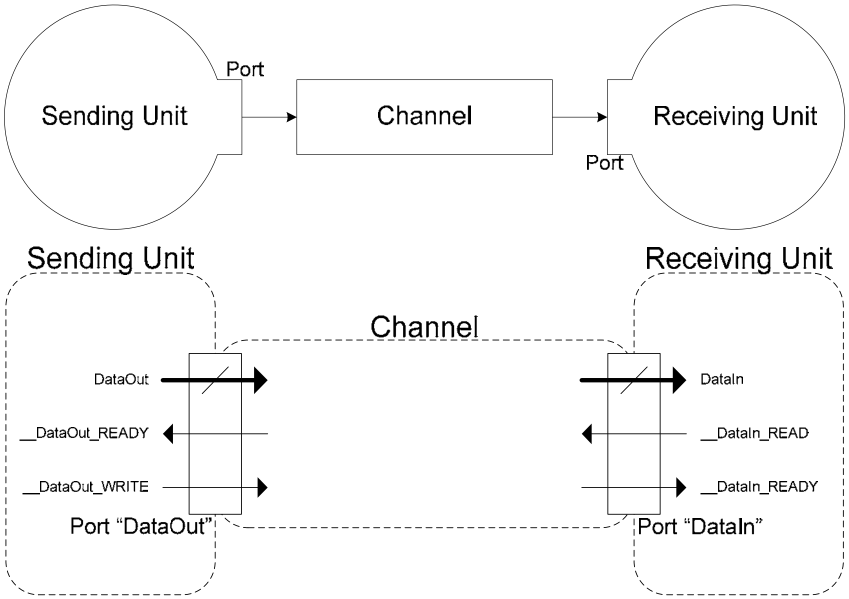
- 32 CPUs each with 32KB L1 data-cache on one BEE2 board
- CPUs, Caches, Links operate at 100MHz
- Shared FPU every 4 CPUs (1 per FPGA)
- Each CPU runs uC Linux (microcontroller Linux)
- CPUs are softcore MicroBlazes (32-bit Xilinx RISC architecture)

RAMP Project Status

- NSF infrastructure grant awarded 3/06
  - 2 staff positions (NSF sponsored), no grad students
- IBM Faculty Awards to RAMPants 6/06
  - Krste Asanovic (MIT), Derek Chiou (Texas), James Hoe (CMU), Christos Kozyrakis (Stanford), John Wawrzynek (Berkeley)
- 3-day retreats with industry visitors
  - "Berkeley-style" retreats 1/06 (Berkeley), 6/06 (ISCA/Boston), 1/07 (Berkeley), 6/07 (ISCA/San Diego)
- RAMP 1/RDL short course
  - 40 people from 6 schools 1/06

RAMP Description Language (RDL)

- RDL describes plumbing to connect units together ≈ “Hardware Scripting Language”
- Design composed of **units** that send messages over **channels** via ports
- Units (10,000 + gates)
  - CPU + L1 cache, DRAM controller…
- Channels (≈ FIFO)
  - Lossless, point-to-point, unidirectional, in-order delivery…
- Generates HDL to connect units



RDL at technological sweet spot

- Matches current chip design style
  - Locally synchronous, globally asynchronous
- To plug unit (in any HDL) into RAMP infrastructure, just add RDL "wrapper"
- Units can also be in C or Java or System C or …
  ⇒ Allows debugging design at high level
- Compiles target interconnect onto RAMP paths
  - Handles housekeeping of data width, number of transfers
- FIFO communication model
  ⇒ Computer can have deterministic behavior
  - Interrupts, memory accesses, … exactly same clock cycle each run

  ⇒ Easier to debug parallel software on RAMP

RDL Developed by Krste Asanovíc and Greg Giebling

Related Approaches

- Quickturn, Axis, IKOS, Thara:
  - FPGA- or special-processor based gate-level hardware emulators
  - Synthesizable HDL is mapped to array for cycle and bit-accurate netlist emulation
  - RAMP's emphasis is on emulating high-level architecture behaviors
    - Hardware and supporting software provides architecture-level abstractions for modeling and analysis
    - Targets architecture and software research
    - Provides a spectrum of tradeoffs between speed and accuracy/precision of emulation
- RPM at USC in early 1990's:
  - Up to only 8 processors
  - Only the memory controller implemented with configurable logic

RAMP's Potential Beyond Manycore

- **Attractive Experimental Systems Platform: Standard ISA + standard OS + modifiable + fast enough + trace/measure anything**
  - Generate Long Traces of Full Systems
  - Test Hardware Security Enhancements
  - Inserting Faults to Test Availability Schemes
  - Test design of switches and routers
  - SW Libraries for 128-bit floating point
  - App-specific instruction extensions (≈Tensilica)
  - Alternative Data Center designs
    - Akamai vs. Google: N centers of M computers

RAMP's Potential to Accelerate MPP

- **With RAMP: Fast, wide-ranging exploration of HW/SW options + head-to-head competitions to determine winners and losers**
  - □ Common artifact for HW and SW researchers ⇒ innovate across HW/SW boundaries
  - □ Minutes vs. years between "HW generations"
  - □ Cheap, small, low power ⇒ Every dept owns one
  - □ FTP supercomputer overnight, check claims locally
  - □ Emulate any MPP ⇒ aid to teaching parallelism
  - □ If IBM, Intel, …had RAMP boxes
    ⇒ Easier to carefully evaluate research claims
    ⇒ Help technology transfer
- **Without RAMP: One Best Shot + Field of Dreams?**

Multiprocessing Watering Hole

Parallel file system Dataflow language/computer Data center in a box
Fault insertion to check dependability Router design Compile to FPGA
Flight Data Recorder Security enhancements Transactional Memory
Internet in a box 128-bit Floating Point Libraries Parallel languages

- Killer app: ≈ All CS Research, Advanced Development
- RAMP attracts many communities to shared artifact
  ⇒ Cross-disciplinary interactions
  ⇒ Ramp up innovation in multiprocessing
- RAMP as next Standard Research/AD Platform? (e.g., VAX/BSD Unix in 1980s)

Supporters and Participants

- Gordon Bell (Microsoft)
- Ivo Bolsens (Xilinx CTO)
- Jan Gray (Microsoft)
- Norm Jouppi (HP Labs)
- Bill Kramer (NERSC/LBL)
- Konrad Lai (Intel)
- Craig Mundie (MS CTO)
- Jaime Moreno (IBM)
- G. Papadopoulos (Sun CTO)
- Jim Peek (Sun)
- Justin Rattner (Intel CTO)
- Michael Rosenfield (IBM)
- Tanaz Sowdagar (IBM)
- Ivan Sutherland (Sun Fellow)
- Chuck Thacker (Microsoft)
- Kees Vissers (Xilinx)
- Jeff Welser (IBM)
- David Yen (Sun EVP)
- *Doug Burger (Texas)*
- *Bill Dally (Stanford)*
- *Susan Eggers (Washington)*
- *Kathy Yelick (Berkeley)*

RAMP Participants: Arvind (MIT), Krste Asanovíc (MIT), Derek Chiou (Texas), James Hoe (CMU), Christos Kozyrakis (Stanford), Shih-Lien Lu (Intel), Mark Oskin (Washington), David Patterson (Berkeley, Co-PI), Jan Rabaey (Berkeley), and John Wawrzynek (Berkeley, PI)

Conclusions

- **Carpe Diem: need RAMP yesterday**
  - □ System emulation + good accounting vs. FPGA computer
  - □ FPGAs ready now, and getting better
  - □ Stand on shoulders vs. toes: standardize on BEE2
  - □ Architects aid colleagues via gateware
- **RAMP accelerates HW/SW generations**
  - □ Emulate, Trace, Reproduce anything; Tape out every day
  - □ RAMP⇒ search algorithm, language ***and*** architecture space
- "**Multiprocessor Research Watering Hole**" Ramp up research in multiprocessing via common research platform ⇒ innovate across fields ⇒ hasten sea change from sequential to parallel computing

Backup Slides

Why RAMP More Believable?

- Starting point for processor is debugged design from Industry in HDL
- HDL units implement operation vs. a high-level description of function
  - Model queuing delays at buffers by building real buffers
- Must work well enough to run OS
  - Can’t go backwards in time, which simulators can
- Can measure anything as sanity checks

Why RAMP Now?

- FPGAs kept doubling resources / 18 months
  - 1994: N FPGAs / CPU, 2005
  - 2006: 256X more capacity ⇒ N CPUs / FPGA
- We are emulating a target system to run experiments, not "just" a FPGA supercomputer
- Given Parallel Revolution, challenges today are organizing large units vs. design of units
- Downloadable IP available for FPGAs
- FPGA design and chip design similar, so results credible when can't fab believable chips

RAMP Development Plan

1. Distribute systems internally for RAMP 1 development
   - Xilinx agreed to pay for production of a set of modules for initial contributing developers and first full RAMP system
   - Others could be available if can recover costs
2. Release publicly available out-of-the-box MPP emulator
   - Based on standard ISA (IBM Power, Sun SPARC, …) for binary compatibility
   - Complete OS/libraries
   - **Locally modify RAMP as desired**
3. Design next generation platform for RAMP 2
   - Base on 65nm FPGAs (2 generations later than Virtex-II)
   - Pending results from RAMP 1, Xilinx will cover hardware costs for initial set of RAMP 2 machines
   - **Find 3rd party to build and distribute systems (at *near-cost*), open source RAMP gateware and software**
   - **Hope RAMP 3, 4, … self-sustaining**

- NSF/CRI proposal pending to help support effort
  - 2 full-time staff (one HW/gateware, one OS/software)
  - Look for grad student support at 6 RAMP universities from industrial donations

RAMP Example: UT FAST

- 1MHz to 100MHz, cycle-accurate, full-system, multiprocessor simulator
  - □ Well, not quite that fast right now, but we are using embedded 300MHz PowerPC 405 to simplify
- X86, boots Linux, Windows, targeting 80486 to Pentium M-like designs
  - □ Heavily modified Bochs, supports instruction trace and rollback
- Working on "superscalar" model
  - □ Have straight pipeline 486 model with TLBs and caches
- Statistics gathered in hardware
  - □ Very little if any probe effect
- Work started on tools to semi-automate micro-architectural and ISA level exploration
  - □ Orthogonality of models makes both simpler

Derek Chiou, UTexas

Example: Transactional Memory

- Processors/memory hierarchy that support transactional memory
- Hardware/software infrastructure for performance monitoring and profiling
  - Will be general for any type of event
- Transactional coherence protocol

Christos Kozyrakis, Stanford

Example: PROTOFLEX

- Hardware/Software Co-simulation/test methodology
- Based on FLEXUS C++ full-system multiprocessor simulator
  - Can swap out individual components to hardware
- Used to create and test a non-block MSI invalidation-based protocol engine in hardware

Example: Wavescalar Infrastructure

- Dynamic Routing Switch
- Directory-based coherency scheme and engine

Example RAMP App: “Enterprise in a Box”

- Building blocks also ⇒ Distributed Computing
- RAMP vs. Clusters (Emulab, PlanetLab)
  - □ Scale: RAMP O(1000) vs. Clusters O(100)
  - □ Private use: $100k ⇒ Every group has one
  - □ Develop/Debug: Reproducibility, Observability
  - □ Flexibility: Modify modules (SMP, OS)
  - □ Heterogeneity: Connect to diverse, real routers
- Explore via repeatable experiments as vary parameters, configurations vs. observations on single (aging) cluster that is often idiosyncratic